MA 3046
Matrix Algebra
Final Exam - Quarter II - AY 03-04

Instructions: Work all problems. Show appropriate intermediate work for full or partial credit. Three pages of notes ($8\frac{1}{2}$ by 11 inches, both sides, handwritten) permitted. *Read the questions carefully.*

---

1. (35 points) Using the **QR** method, solve the system:

$$\mathbf{A}\,\mathbf{x} = \begin{bmatrix} 2 & 1 \\ 2 & -7 \\ 2 & 1 \\ 2 & -7 \end{bmatrix} \mathbf{x} = \begin{bmatrix} -2 \\ 14 \\ 12 \\ 12 \end{bmatrix}$$

solution:

In order to solve this by the **QR** method, we must first find the **QR** factorization of **A**. We could do this by any of several methods, but Gram-Schmidt is probably easiest. (Note that, in this case, with only two columns, the classic and modified versions are identical. Also note that, because the original matrix **A** is only 4×2, this problem is likely only solvable in the least-squares sense.)

The modified Gram-Schmidt algorithm is:

(1) Form: $\mathbf{v}^{(j)} = \mathbf{a}^{(j)}$, $j = 1, 2, \ldots, n$

(2) For $j = 1, 2, \ldots, n-1$:

Form: $r_{jj} = \| \mathbf{v}^{(j)} \|$ and $\mathbf{q}^{(j)} = \mathbf{v}^{(j)}/r_{jj}$

For $k = (j+1), \cdots, n$, do

$$r_{jk} = {\mathbf{q}^{(j)}}^H \mathbf{v}^{(k)}$$
$$\mathbf{v}^{(k)} = \mathbf{v}^{(k)} - r_{jk}\mathbf{q}^{(j)}$$

(3) Finally, form: $r_{nn} = \| \mathbf{v}^{(n)} \|$ and $\mathbf{q}^{(n)} = \mathbf{v}^{(n)}/r_{nn}$

So, in this problem, we start with

$$\mathbf{v}^{(1)} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \end{bmatrix} \quad \text{and} \quad \mathbf{v}^{(2)} = \begin{bmatrix} 1 \\ -7 \\ 1 \\ -7 \end{bmatrix}$$

solution:

Therefore, for j=1: $r_{11} = \| \mathbf{a}^{(1)} \| = \sqrt{2^2 + 2^2 + 2^2 + 2^2} = \sqrt{16} = 4$, and so

$$\mathbf{q}^{(1)} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{bmatrix}$$

Proceeding then to remove the components in this direction from the remaining vectors, we have, for $k = 2$,

$$r_{12} = \mathbf{q}^{(1)^H} \mathbf{v}^{(2)} = [\ \tfrac{1}{2} \quad \tfrac{1}{2} \quad \tfrac{1}{2} \quad \tfrac{1}{2}\ ] \begin{bmatrix} 1 \\ -7 \\ 1 \\ -7 \end{bmatrix} = -6$$

and so

$$\mathbf{v}^{(2)} = \mathbf{v}^{(2)} - (1)\mathbf{q}^{(1)} = \begin{bmatrix} 1 \\ -7 \\ 1 \\ -7 \end{bmatrix} - (-6) \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{bmatrix} = \begin{bmatrix} 4 \\ -4 \\ 4 \\ -4 \end{bmatrix}$$

Next, for j = 2,

$$r_{22} = \| \mathbf{v}^{(2)} \| = 8 \quad \text{and so} \quad \mathbf{q}^{(2)} = \mathbf{v}^{(2)}/8 = \begin{bmatrix} \frac{1}{2} \\ -\frac{1}{2} \\ \frac{1}{2} \\ -\frac{1}{2} \end{bmatrix}$$

solution:

Therefore, the **QR** decomposition of the original matrix is:

$$\begin{bmatrix} 2 & 1 \\ 2 & -7 \\ 2 & 1 \\ 2 & -7 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} 4 & -6 \\ 0 & 8 \end{bmatrix}$$

Then, since $\mathbf{Q}^H \mathbf{Q} = \mathbf{I}$, the solution of

$$\mathbf{A}\,\mathbf{x} = \mathbf{Q}\,\mathbf{R}\,\mathbf{x} = \mathbf{b}$$

is obtained by solving

$$\mathbf{R}\,\mathbf{x} = \mathbf{Q}^H\,\mathbf{b}$$

i.e.

$$\begin{bmatrix} 4 & -6 \\ 0 & 8 \end{bmatrix} \mathbf{x} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} -2 \\ 14 \\ 12 \\ 12 \end{bmatrix} = \begin{bmatrix} 18 \\ -8 \end{bmatrix}$$

or

$$\begin{array}{rcrcr} 4x_1 & - & 6x_2 & = & 18 \\ & & 8x_2 & = & -8 \end{array} \quad \Longrightarrow \quad \begin{array}{rcr} x_1 & = & 3 \\ x_2 & = & -1 \end{array}$$

Note, although not required, this solution can easily be checked. But, in doing so, it is **vital** to remember this is a least-squares problem! We can easily show that

$$\begin{bmatrix} 2 & 1 \\ 2 & -7 \\ 2 & 1 \\ 2 & -7 \end{bmatrix} \begin{bmatrix} 3 \\ -1 \end{bmatrix} = \begin{bmatrix} 5 \\ 13 \\ 5 \\ 13 \end{bmatrix} \neq \mathbf{b}$$

but

$$\mathbf{r} = \mathbf{b} - \mathbf{A}\,\mathbf{x} = \begin{bmatrix} -2 \\ 14 \\ 12 \\ 12 \end{bmatrix} - \begin{bmatrix} 5 \\ 13 \\ 5 \\ 13 \end{bmatrix} = \begin{bmatrix} -7 \\ 1 \\ 7 \\ -1 \end{bmatrix}$$

which is obviously orthogonal to both columns of **A**. Therefore we have the correct least-squares solution.

2. (40 points) a. Using partial pivoting, and simulating a three-digit decimal computer that rounds all intermediate calculations, complete the partial $\mathbf{P}\,\mathbf{A} = \mathbf{L}\,\mathbf{U}$ decomposition shown (note no row interchanges have been required up to this point):

$$\mathbf{A} = \begin{bmatrix} 2 & -2 & 4 & 2 \\ -1 & 2 & -5 & 0 \\ 1 & 2 & -9 & 8 \\ 1 & 1 & -2 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -.5 & 1 & 0 & 0 \\ .5 & 0 & 1 & 0 \\ .5 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 4 & 2 \\ 0 & 1 & -3 & 1 \\ 0 & 3 & -11 & 7 \\ 0 & 2 & -4 & -2 \end{bmatrix}$$

solution:

Observe that, at this point, we have

$$\mathbf{L}_{work} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -.5 & 1 & 0 & 0 \\ .5 & 0 & 1 & 0 \\ .5 & 0 & 0 & 1 \end{bmatrix} , \; \mathbf{U}_{work} = \begin{bmatrix} 2 & -2 & 4 & 2 \\ 0 & 1 & -3 & 1 \\ 0 & 3 & -11 & 7 \\ 0 & 2 & -4 & -2 \end{bmatrix}$$

and $\qquad \mathbf{p} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$

But now the largest element in the working portion of the second column is on the third row. So we must interchange

(1) The second and third rows of $\mathbf{U}_{work}$.
(2) The subdiagonal elements in the second and third rows of $\mathbf{L}_{work}$.
(3) The second and third rows of $\mathbf{p}$. to give

$$\mathbf{L}_{work} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ .5 & 1 & 0 & 0 \\ -.5 & 0 & 1 & 0 \\ .5 & 0 & 0 & 1 \end{bmatrix} , \quad \mathbf{U}_{work} = \begin{bmatrix} 2 & -2 & 4 & 2 \\ 0 & 3 & -11 & 7 \\ 0 & 1 & -3 & 1 \\ 0 & 2 & -4 & -2 \end{bmatrix}$$

and $\qquad \mathbf{p} = \begin{bmatrix} 1 \\ 3 \\ 2 \\ 4 \end{bmatrix}$

solution:

Then we can eliminate in the second column of $\mathbf{U}_{work}$. Emulating a three-digit, rounding machine, this means

$$\begin{bmatrix} 2 & -2 & 4 & 2 \\ 0 & 3 & -11 & 7 \\ 0 & 1 & -3 & 1 \\ 0 & 2 & -4 & -2 \end{bmatrix} \quad \begin{matrix} \\ \\ R_3 - (.333)R_2 \\ R_4 - (.667)R_2 \end{matrix} \quad \Longrightarrow \quad \begin{bmatrix} 2 & -2 & 4 & 2 \\ 0 & 3 & -11 & 7 \\ 0 & 0 & 0.660 & -1.33 \\ 0 & 0 & 3.34 & -6.67 \end{bmatrix}$$

Note we have to be a little "delicate" here to accurately simulate the specified machine. For example, to update the element in the $(3,3)$ position, we should compute:

$$-3.00 - \overbrace{(.333) * (-11.0)}^{-3.663} = -3.00 + 3.66 = .660$$

After we also update the corresponding elements of $\mathbf{L}_{work}$, we have:

$$\mathbf{L}_{work} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ .5 & 1 & 0 & 0 \\ -.5 & 0.333 & 1 & 0 \\ .5 & 0.667 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{U}_{work} = \begin{bmatrix} 2 & -2 & 4 & 2 \\ 0 & 3 & -11 & 7 \\ 0 & 0 & 0.660 & -1.33 \\ 0 & 0 & 3.34 & -6.67 \end{bmatrix}$$

Next we must eliminate in the third column. But, again, the largest element in the working portion of that column is not on the diagonal. So we must first interchange :

(1) The third and fourth rows of $\mathbf{U}_{work}$.

(2) The subdiagonal elements in the third and fourth rows of $\mathbf{L}_{work}$.

(3) The third and fourth rows of $\mathbf{p}$.

This yields:

$$\mathbf{L}_{work} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ .5 & 1 & 0 & 0 \\ .5 & 0.667 & 1 & 0 \\ -.5 & 0.333 & 0 & 1 \end{bmatrix}, \quad \mathbf{U}_{work} = \begin{bmatrix} 2 & -2 & 4 & 2 \\ 0 & 3 & -11 & 7 \\ 0 & 0 & 3.34 & -6.67 \\ 0 & 0 & 0.660 & -1.33 \end{bmatrix}$$

and $\quad \mathbf{p} = \begin{bmatrix} 1 \\ 3 \\ 4 \\ 2 \end{bmatrix}$

solution:

Now we can proceed with elimination:

$$\begin{bmatrix} 2 & -2 & 4 & 2 \\ 0 & 3 & -11 & 7 \\ 0 & 0 & 3.34 & -6.67 \\ 0 & 0 & 0.660 & -1.33 \end{bmatrix} \begin{matrix} \\ \\ \\ R_4 - (0.198)R_3 \end{matrix} \Longrightarrow \begin{bmatrix} 2 & -2 & 4 & 2 \\ 0 & 3 & -11 & 7 \\ 0 & 0 & 3.34 & -6.67 \\ 0 & 0 & 0 & -0.0100 \end{bmatrix}$$

and so now we can fill in the final element in

$$\mathbf{L}_{work} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ .5 & 1 & 0 & 0 \\ .5 & 0.667 & 1 & 0 \\ -.5 & 0.333 & 0.198 & 1 \end{bmatrix}$$

Therefore $\mathbf{P}\,\mathbf{A} = \mathbf{L}\,\mathbf{U}$ where

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad , \quad \mathbf{L} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ .5 & 1 & 0 & 0 \\ .5 & 0.667 & 1 & 0 \\ -.5 & 0.333 & 0.198 & 1 \end{bmatrix}$$

and

$$\mathbf{U} = \begin{bmatrix} 2 & -2 & 4 & 2 \\ 0 & 3 & -11 & 7 \\ 0 & 0 & 3.34 & -6.67 \\ 0 & 0 & 0 & -0.0100 \end{bmatrix}$$

b. Based on your computations in part a, do you think this matrix is well-conditioned for a three digit machine?

solution:

Despite the use of partial pivoting, we still have a "small" (order of magnitidue of three-digit machine precision) pivot in **L**. Since a zero pivot would connote a singular matrix, this small pivot implies that **A** is nearly singular. Therefore, we expect this matrix to be ill-conditioned in a three-digit machine.

3. (30 points) Consider the matrix:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & -3 \\ -1 & 5 & 1 \\ 2 & 3 & -1 \end{bmatrix}$$

Five iterations of the power method *without normalization after each step* have produced:

$$\mathbf{x}^{(5)} = \begin{bmatrix} 457 \\ 4520 \\ 2315 \end{bmatrix}$$

Conduct one more iteration of the method, and estimate both the dominant eigenvalue and its associated eigenvector.

solution:

Although not necessary (since eigenvectors are unique only up to direction), we will normalize at this point, using the infinity norm:

$$\mathbf{x}^{(5)} = \frac{\mathbf{x}^{(5)}}{4520} = \begin{bmatrix} 0.1011 \\ 1.0000 \\ 0.5122 \end{bmatrix}$$

Now do one more iteration of the power method:

$$\mathbf{x}^{(6)} = \mathbf{A}\,\mathbf{x}^{(5)} = \begin{bmatrix} 1 & 2 & -3 \\ -1 & 5 & 1 \\ 2 & 3 & -1 \end{bmatrix} \begin{bmatrix} 0.1011 \\ 1.0000 \\ 0.5122 \end{bmatrix} = \begin{bmatrix} 0.5646 \\ 5.4111 \\ 2.6900 \end{bmatrix}$$

At this point, we may or may not normalize again. We choose to:

$$\mathbf{x}^{(6)} = \frac{\mathbf{x}^{(6)}}{5.4111} = \begin{bmatrix} 0.1043 \\ 1.0000 \\ 0.4971 \end{bmatrix}$$

The eigenvalue is now best estimated using the Rayleigh quotient:

$$R = \frac{\mathbf{x}^{(6)^T}\,\mathbf{A}\,\mathbf{x}^{(6)}}{\mathbf{x}^{(6)^T}\,\mathbf{x}^{(6)}}$$

Note

$$\mathbf{A}\,\mathbf{x}^{(6)} = \begin{bmatrix} 0.6129 \\ 5.3928 \\ 2.7115 \end{bmatrix} \qquad \Longrightarrow \qquad \mathbf{x}^{(6)^T}\,\mathbf{A}\,\mathbf{x}^{(6)} = 6.8048$$

solution:

and

$$\mathbf{x}^{(6)^T}\mathbf{x}^{(6)} = 1.2580 \quad \Longrightarrow \quad R = \frac{6.8048}{1.2580} = 5.4090$$

Therefore

$$\lambda_1 = 5.4090 \quad \text{and} \quad \mathbf{q}^{(1)} = \begin{bmatrix} 0.1043 \\ 1.0000 \\ 0.4971 \end{bmatrix}$$

4. (35 points) a. Perform *two* iterations of the *Gauss-Seidel* method for the solution of:

$$\begin{bmatrix} 8 & 1 & -2 \\ 0 & 4 & 1 \\ -2 & 0 & 10 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 4 \\ 1 \\ -9 \end{bmatrix}$$

starting with

$$\mathbf{x}^{(0)} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

solution:

The Gauss-Seidel algorithm for this system can be formulated by:
(i) Moving the off-diagonal terms to the right-hand side of the equation,
(ii) Dividing each equation by the diagonal coefficient
(iii) Replacing the above-diagonal terms on the the right by their values from the previous iteration, and
(iv) Replacing the below-diagonal terms on the right by their values from the current iteration

i.e., for this system:

$$\begin{aligned} x_1^{(k+1)} &= && - \tfrac{1}{8}x_2^{(k)} + \tfrac{1}{4}x_3^{(k)} + \tfrac{1}{2} \\ x_2^{(k+1)} &= && - \tfrac{1}{4}x_3^{(k)} + \tfrac{1}{4} \\ x_3^{(k+1)} &= \tfrac{1}{5}x_1^{(k+1)} && - \tfrac{9}{10} \end{aligned}$$

Proceeding

$$\begin{aligned} x_1^{(1)} &= && - \tfrac{1}{8}x_2^{(0)} + \tfrac{1}{4}x_3^{(0)} + \tfrac{1}{2} \\ x_2^{(1)} &= && - \tfrac{1}{4}x_3^{(0)} + \tfrac{1}{4} \\ x_3^{(1)} &= \tfrac{1}{5}x_1^{(1)} && - \tfrac{9}{10} \end{aligned}$$

or

$$\begin{aligned} x_1^{(1)} &= && - \tfrac{1}{8}(0) + \tfrac{1}{4}(0) + \tfrac{1}{2} &&= 0.5000 \\ x_2^{(1)} &= && - \tfrac{1}{4}(0) + \tfrac{1}{4} &&= 0.2500 \\ x_3^{(1)} &= \tfrac{1}{5}(0.5000) && - \tfrac{9}{10} &&= -0.8000 \end{aligned}$$

solution:

or

$$\mathbf{x}^{(1)} = \begin{bmatrix} 0.5000 \\ 0.2500 \\ -0.8000 \end{bmatrix}$$

For the next iteration:

$$\begin{aligned}
x_1^{(2)} &= & & - \tfrac{1}{8}x_2^{(1)} + \tfrac{1}{4}x_3^{(1)} + \tfrac{1}{2} \\
x_2^{(2)} &= & & - \tfrac{1}{4}x_3^{(1)} + \tfrac{1}{4} \\
x_3^{(2)} &= \tfrac{1}{5}x_1^{(2)} & & - \tfrac{9}{10}
\end{aligned}$$

or

$$\begin{aligned}
x_1^{(2)} &= & & - \tfrac{1}{8}(0.2500) + \tfrac{1}{4}(-0.8000) + \tfrac{1}{2} &&= 0.2687 \\
x_2^{(2)} &= & & - \tfrac{1}{4}(-0.8000) + \tfrac{1}{4} &&= 0.4500 \\
x_3^{(2)} &= \tfrac{1}{5}(0.2687) & & - \tfrac{9}{10} &&= -0.8463
\end{aligned}$$

Therefore

$$\mathbf{x}^{(2)} = \begin{bmatrix} 0.2687 \\ 0.4500 \\ -0.8463 \end{bmatrix}$$

(Note the exact solution is:

$$\mathbf{x} = \begin{bmatrix} 0.228476\ldots \\ 0.463576\ldots \\ -0.854304\ldots \end{bmatrix}$$

and so our iterative solution is already not to bad.

b. Was Gauss-Seidel a "good" choice for this problem? *Briefly* explain your answer.

solution:

Probably not, at least assuming that the criteria for "best" require finding a reasonably correct solution (effectiveness) with the minimum number of computations (efficiency). This is neither a large nor a sparse problem. Gaussian Elimination would get the **exact** solution in about nineteen flops. These two iterations have already cost about twenty-one flops, and so far we only have answers that are accurate to one to two significant digits. The mere fact that **A** is diagonally dominant here, and therefore convergence is guaranteed, does not alone make an iterative method a "good" choice unless your explicit criterion for good is that the algorithm produces a reasonably correct solution irrespective of cost.

5. (30 points) Consider the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ \epsilon & 0 \\ 0 & \epsilon \end{bmatrix}$$

a. Show that the singular values of this matrix are exactly $\sigma_1 = \sqrt{2+\epsilon^2}$ and $\sigma_2 = \epsilon$. (Do **not** do the entire singular value decomposition!)

solution:

By definition, the singular values of $\mathbf{A}$ are the square roots of the eigenvalues of $\mathbf{A}^H\,\mathbf{A}$. So first compute

$$\mathbf{A}^H\,\mathbf{A} = \begin{bmatrix} 1 & \epsilon & 0 \\ 1 & 0 & \epsilon \end{bmatrix} \begin{bmatrix} 1 & 1 \\ \epsilon & 0 \\ 0 & \epsilon \end{bmatrix} = \begin{bmatrix} 1+\epsilon^2 & 1 \\ 1 & 1+\epsilon^2 \end{bmatrix}$$

Then, if σ_i are the singular values of $\mathbf{A}$, the eigenvalues of this matrix must be $\lambda_1 = 2+\epsilon^2$ and $\lambda_2 = \epsilon^2$. So check:

$$\mathbf{A}^H\,\mathbf{A} - (2+\epsilon^2)\mathbf{I} == \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$

which is clearly singular. Similarly,

$$\mathbf{A}^H\,\mathbf{A} - \epsilon^2\mathbf{I} == \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

which is also obviously singular. Therefore the given values are the singular values of $\mathbf{A}$.

b. Suppose ϵ is a sufficiently small number that, in some computers, because of round-off errors, the quantity $1+\epsilon^2$ actually evaluates to 1. How do the numerically-computed singular values then differ from the actual ones.

solution:

In this case,

$$fl\left(\mathbf{A}^H\,\mathbf{A}\right) = fl\left(\begin{bmatrix} 1+\epsilon^2 & 1 \\ 1 & 1+\epsilon^2 \end{bmatrix}\right) = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

solution:

Since

$$\det\left(fl\left(\mathbf{A}^H\,\mathbf{A}\right) - \lambda\mathbf{I}\right) = \det\begin{bmatrix} 1-\lambda & 1 \\ 1 & 1-\lambda \end{bmatrix} = \lambda^2 - 2\lambda$$

The eigenvalues of this matrix are easily shown to be

$$\tilde{\lambda}_1 = 2 \quad \text{and} \quad \tilde{\lambda}_2 = 0$$

Therefore the computed singular values will be:

$$\tilde{\sigma}_1 = \sqrt{2} \quad \text{and} \quad \tilde{\sigma}_2 = 0$$

In otherwords, this matrix is *numerically singular*.

c. What is the actual condition number of $\mathbf{A}$ (in the Euclidean norm). Based on this result, is the result you obtained in part b. above reasonable?

solution:

The condition number of the original matrix, in the Euclidean norm, is given by:

$$\kappa(\mathbf{A}) = \frac{\sigma_1}{\sigma_2} = \frac{\sqrt{2+\epsilon^2}}{\epsilon}$$

From this, it is obvious that, for “small” ϵ,

$$\kappa(\mathbf{A}) \doteq \frac{\sqrt{2}}{\epsilon}$$

Therefore, the matrix will be very ill-conditioned, i.e. nearly singular in this case. Therefore, the fact that round-off errors can make it exactly singular should not be that surprising.

6. (30 points) A 5000×1 vector $\mathbf{x}$ must undergo a projection given by:

$$\left(\mathbf{I} - \mathbf{P}\,\mathbf{P}^H\right)\,\mathbf{x}$$

where $\mathbf{P}$ is a 5000×3 matrix, the first fourty-two hundred rows of which are *identically zero*. The result will be stored in the a new location associated with the vector $\mathbf{y}$. Give no more than four lines of MATLAB code that will accomplish this in a highly efficient manner. Estimate the number of flops and amount of additional storage required by your code.

solution:

Note that if we, conceptually, partition $\mathbf{P}$ and $\mathbf{x}$ as follows:

$$\mathbf{P} = \begin{bmatrix} \mathbf{0} \\ \cdots \\ \mathbf{P}_{21} \end{bmatrix} \quad \text{and} \quad \mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \cdots \\ \mathbf{x}_2 \end{bmatrix}$$

where $\mathbf{P}_{21}$ is 800×3, etc., then we see that:

$$\mathbf{y} = \left(\mathbf{I} - \mathbf{P}\,\mathbf{P}^H\right)\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \cdots \\ \mathbf{x}_2 \end{bmatrix} - \begin{bmatrix} \mathbf{0} \\ \cdots \\ \mathbf{P}_{21} \end{bmatrix} \left(\begin{bmatrix} \mathbf{0} & \vdots & \mathbf{P}_{21}^H \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \cdots \\ \mathbf{x}_2 \end{bmatrix} \right)$$

$$= \begin{bmatrix} \mathbf{x}_1 \\ \cdots \\ \mathbf{x}_2 \end{bmatrix} - \begin{bmatrix} \mathbf{0} \\ \cdots \\ \mathbf{P}_{21} \end{bmatrix} \left({\mathbf{P}_{21}}^H\,\mathbf{x}_2 \right)$$

$$= \begin{bmatrix} \mathbf{x}_1 \\ \cdots\cdots\cdots\cdots \\ \mathbf{x}_2 - \mathbf{P}_{21}\left({\mathbf{P}_{21}}^H\,\mathbf{x}_2\right) \end{bmatrix}$$

Probably the most efficient MATLAB code for this operation would be

```
y              = x
v              = P(4201:5000,4201:5000)'*x(4201:5000)
y(4201:5000)   = y(4201:5000) ...
                    - P(4201:5000,4201:5000)*v
```

solution:

Implementing this method will require:

(1) Multiplying one 3×800 matrix ($\mathbf{P}_{21}^H$) by an 800×1 vector ($\mathbf{x}_2$), at a cost of approximately 4800 ($= 2 \times 800 \times 3$) flops, plus

(2) Multiplying the resulting 3×1 vector on the left by an 800×3 matrix ($\mathbf{P}_{21}$, at the cost of another approximately 4800 flops, then finally

(3) Subtracting two 800×1 vectors, at the cost of 800 flops.

Total cost $=$ 10,400 flops (approximately).

This code would requie, besides the 5000 location needed to hold $\mathbf{y}$, an additional 800 storage locations to hold $\mathbf{v}$, plus and up to an additional 800 temporary storage locations to hold $\mathbf{P}_{21}\mathbf{v}$. This cost is negligible.